Holly: A Drawing Editor for Stencil Design

Yuki Igarashi*
The University of Tokyo

Takeo Igarashi
The University of Tokyo / JST ERATO

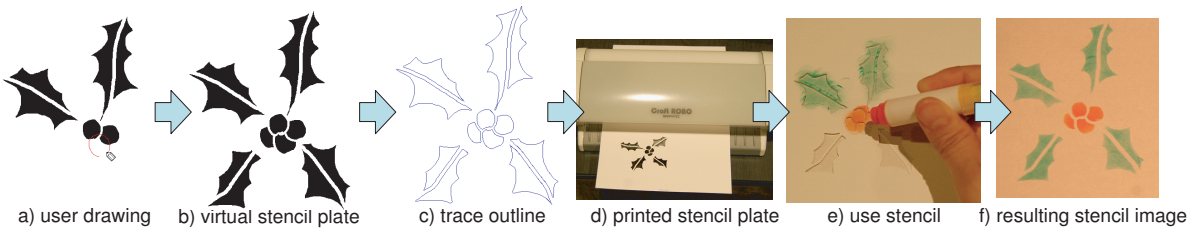

Figure 1: *Overview of our system. A user draws freeform strokes. The system automatically generates a virtual stencil plate. The user designs a physical stencil plate using a cutting plotter. Finally, the user adds pigment using the stencil plate.*

1 Introduction

Stencil is a form of artistic expression that prints special pattern on a target object by applying pigment over a plate with holes (Fig. 1 f). A stencil plate can be characterized as containing two types of regions, positive and negative. A negative region is an empty space through which paint, while a positive region is an entirely connected surface of material. Bronson et al. [2008] presented a method for generating expressive stencils from 3D polygonal meshes or images. In contrast, our goal was to design a stencil plate from scratch using drawing operations.

2 User Interface

Fig. 1 shows the overall process of our system. The user interactively draws free-form strokes on the canvas as in standard drawing editors. Our current implementation supports basic primitives such as brush strokes, fill stroke, erasing strokes. It also allows the user to edit these primitives such as to move, delete, change order, copy and paste them. The system then automatically generates stencil plate image in which all positive regions are connected (Fig. 1 b). Our system avoids the isolation of positive regions by always placing positive margin around the negative region, but it can still cause an isolated positive region surrounded by negative region. In this case, the system automatically adds a bridge to resolve the isolation. The system then automatically generates a valid stencil plate (Fig. 1 c). The user can print out the final image using a cutting plotter. Finally the user applies pigment over the stencil plate on a target material. We demonstrate that non-professional users can design stencil plates using our system.

3 Implementation

Our prototype system is implemented as a Java™ program, and outputs stencil outlines in DXF vector file. The system runs in real-time on a 1.1 GHz Pentium M PC. Internally, the system uses a hybrid representation combining vector and raster representation. The user's strokes are stored as vector graphics primitives. The system then renders them onto a raster image and executes Boolean operations and clustering on the raster image. The system puts margin around strokes seen in Fig. 2 by first drawing a positive stroke with extra margin before drawing a negative stroke on top of it. During drawing and erasing, the stencil image sometimes includes an isolated positive region, which is not allowed in a stencil. The system automatically detects the isolated region (we call an island) and adds bridges (Fig. 3). The system first applies flood fill algorithm repeatedly to the stencil image in order to decompose the positive pixels into connected regions. The system then computes distances between each pair of connected regions and a minimum spanning tree of the graph. The system places bridges at the edges contained in the tree. The bridges are represented as ephemeral primitives. They are deleted and re-computed every time the user modifies the image. The system finally traces the boundary of negative regions (Fig. 1 c) and exports the result in a vector form (we support SVG and DXF format).

Figure 2: *Stencil plate images created using our system, and real results.*

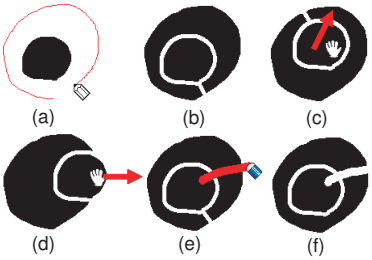

Figure 3: *The system automatically detects islands and adds a bridge to connect it to the main sheet.*

References

BRONSON, J., RHEINGANS, P., AND OLANO, M. 2008. Semi-automatic stencil creation through error minimization. In *Non-Photorealistic Animation and Rendering Symposium*, 31–37.

*e-mail:{yukim, takeo}@acm.org

NPAR 2009, New Orleans, Louisiana, August 1 – 2, 2009.
ISBN 978-1-60558-605-2/09/0008